Pooling Design and Bias Correction in DNA Library Screening

Takafumi Kanamori, *Nagoya University,*
Furocho, Chikusaku, Nagoya 464-8603, Japan.
Email: kanamori@is.nagoya-u.ac.jp

Hiroaki Uehara, *Nagoya University,*
Furocho, Chikusaku, Nagoya 464-8603, Japan.
Email: uehara@jim.math.cm.is.nagoya-u.ac.jp

Masakazu Jimbo, *Nagoya University,*
Furocho, Chikusaku, Nagoya 464-8603, Japan.
Email: jimbo@is.nagoya-u.ac.jp

Abstract

We study the group test for DNA library screening based on probabilistic approach. Group test is a method of detecting a few positive items from among a large number of items, and has wide range of applications. In DNA library screening, positive item corresponds to the clone having a specified DNA segment, and it is necessary to identify and isolate the positive clones for compiling the libraries. In the group test, a group of items, called *pool*, is assayed in a lump in order to save the cost of testing, and positive items are detected based on the observation from each pool. It is known that the design of grouping, that is, pooling design is important to achieve accurate detection. In the probabilistic approach, positive clones are picked up based on the posterior probability. Naive methods of computing the posterior, however, involves exponentially many sums, and thus we need a device. Loopy belief propagation (loopy BP) algorithm is one of popular methods to obtain approximate posterior probability efficiently. There are some works investigating the relation between the accuracy of the loopy BP and the pooling design. Based on these works, we develop pooling design with small estimation bias of posterior probability, and we show that the balanced incomplete block design (BIBD) has nice property for our purpose. Some numerical experiments show that the bias correction under the BIBD is useful to improve the estimation accuracy.

Keywords: Group test; Pooling design; Loopy belief propagation; BIB Design.

1 Introduction

We study the group test based on a probabilistic approach. Group test is a method of detecting positive items out of a set of a large number of items, and has wide range of applications such as blood test or DNA library screening.

In the context of DNA library screening, our purpose is to identify clones having a specified DNA fragment from among a collection of DNA segments. Each DNA segment is called clones. The clone with a specified segment is referred to as *positive* clone, otherwise *negative* clone. For large libraries, it is impractical to screen each clone individually, instead a group of clones, called *pool*, is assayed in a lump. This is said to be group test or pooling experiment. When a pool gives positive result, the pool contains at least one positive clone, and otherwise all clones are negative. A number of pools are prepared, and outcomes from all pools are assembled to identify positive clones.

There are mainly two categories of group test; one is adaptive, and the other is non-adaptive. In adaptive strategy, the pool is sequentially prepared and the test is conducted based on the information of previous outcomes. By repeating the test procedure, we can narrow down the set of positive clones. In non-adaptive testing, we prepare all pools to be tested before conducting the group test. The positive clones are detected based on the outcome of each pool. That is, the grouping of clones does not depend on the result of previous testing. When the group test for each pool is performed by distinct experimenters, non-adaptive method may not be time-consuming compared to adaptive one. In this article, we focus on non-adaptive testing.

In group testing, we have two kind of detecting procedure; one is combinatorial and the other is probabilistic. In combinatorial group testing, the main issue is to construct the design of grouping or *pooling design* to reduce the number of testing without missing the positive clones. Combinatorial group testing has been studied by many authors (Du and Hwang, 1999; Ngo and Du, 2000; Wu et al., 2004). In combinatorial approach, it is often assumed that the maximum number of positive clones is known and that there is no observation errors or noisy measurements. On the other hand, in probabilistic approach the prior probability for the state of clones is assumed, and posterior probability such that each clone is positive is computed based on the observation of each pool (Knill et al., 1996; Bruno et al., 1995; Mézard and Toninelli, 2007; Uehara and Jimbo, 2009). The main issue is to develop efficient algorithm to compute the posterior probability, since using naive Bayes formula is computationally demanding. Knill et al. (1996) and Uehara and Jimbo (2009) have proposed a probabilistic algorithm. Knill et al. (1996) have used the Markov Chain Monte Carlo (MCMC) method to obtain the marginal posterior probability, and Uehara and Jimbo (2009) have exploited the *loopy belief propagation* (BP) algorithm (Pearl, 1988; MacKay, 1999) to compute approximate probability.

Non-adaptive group test with probabilistic approach will be one of the most practical methods to detect positive clones from among large DNA library. Even in probabilistic approach, the pooling design is significant to achieve highly accurate estimation of posterior probability. In loopy BP algorithm for the low density parity check (LDPC) coding (MacKay, 1999; Richardson et al., 2001), it has been revealed that the coding design is closely related to the decoding error of the transmitted code. Likewise, the pooling design with some nice property will provide accurate estimator of the posterior probability as experimentally shown by Uehara and Jimbo (2009). In coding theory, Ikeda et al. (2004a) have analyzed the relation between the coding design and the bias of the estimated posterior probability. We apply their result to improve the accuracy of the group testing.

The outline of the paper is as follows. In Section 2 probabilistic description of group testing for DNA library screening is presented. In Section 3 we introduce loopy belief propagation algorithm, and in Section 4 we show the bias the estimated posterior probability according to Ikeda et al. (2004a). In Section 5, we construct a pooling design resulting in a small bias. Numerical experiments are presented in Section 6. Section 7 is devoted to concluding remarks.

2 Preliminaries of DNA library screening

On DNA library screening, our purpose is to identify the positive clones out of a large DNA library. Let X_i be the random variable which stands for the label of the clone i for $i = 1, \ldots, n$, that is, $X_i = 1$ for positive and $X_i = 0$ for negative. The labels of all clones are denoted as the vector $X = (X_1, \ldots, X_n)$. We assume that the random variables X_i, $i = 1, \ldots, n$ are independent. The probability such that $X = x \in \{0,1\}^n$ for $x = (x_1, \ldots, x_n)$ is denoted by $p(x)$ or $p(X = x)$. Then, the probability $p(x)$ is represented by the factorization of marginal probabilities, that is

$$p(x) = p_1(x_1) \times \cdots \times p_n(x_n).$$

Since the marginal distribution $p_i(x_i)$ over $\{0,1\}$ is written as the form of exponential model $p_i(x_i) \propto \exp\{h_i x_i\}$, the joint probability $p(x)$ is given as

$$p(x) = \exp\{h^\top x - \psi_0(h)\}, \quad x \in \{0,1\}^n$$

with $h = (h_1, \ldots, h_n) \in \mathbb{R}^n$, where $\psi_0(h)$ is the normalization factor called the cumulant generating function.

In the group test a number of clones are set in a pool and the experiment is conducted to detect if a positive close is included in the pool. Here the pool is identified by a subset of $\{1, \ldots, n\}$, and the clone i is included in the pool r if and only if $i \in r$ holds. For the pool $r \subset \{1, \ldots, n\}$, let Z_r be the random variable defined by

$$Z_r = \begin{cases} 1 & {}^\exists i \in r,\ X_i = 1, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Hence if $Z_r = 1$, there is a positive clone in the pool r. Note that Z_r is also represented as

$$Z_r = \max_{i \in r} X_i = 1 - \prod_{i \in r}(1 - X_i).$$

In practice, Z_r is not directly observed. The observation of the pool r is usually represented by four levels such that

$$S_r = \begin{cases} 0 & \text{if the pool } r \text{ is negative,} \\ 1 & \text{if the pool } r \text{ is weak positive,} \\ 2 & \text{if the pool } r \text{ is medium positive,} \\ 3 & \text{if the pool } r \text{ is strong positive.} \end{cases}$$

The response of the experiment is measured by using a fluorescence sign, and it is experimentally-confirmed that the conditional probability of S_r given $X_i, (i \in r)$ only depends on Z_r, not the number of $i \in r$ such that $X_i = 1$. We assume that the conditional probability of S_r given Z_r is the same for all pools. Then, the conditional probability of $S_r = s_r$ given $Z_r = z_r$ is denoted as $p(S_r = s_r | Z_r = z_r)$ or $p(s_r|z_r)$. In practice $p(S_r = 0|Z_r = 0)$ and $p(S_r = 3|Z_r = 1)$ will take larger value than others. In the group test usually we prepare a number of pools. Let $\mathcal{G} = \{r_1, \ldots, r_m\}$ be the set of pools used in the group test. Then for each pool $r \in \mathcal{G}$ the observation $s_r \in \{0, 1, 2, 3\}$ is obtained. An example of a pooling design is shown in Figure 1.

The problem considered in the paper is to infer the label of clones based on the observation from each pool. More precisely, we want to pick up only positive clones out of all clones. As a probabilistic

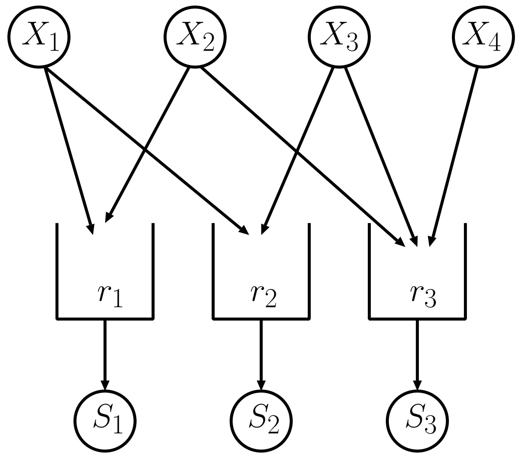

Figure 1: An example of a pooling design. $\mathcal{G}$ is given as $\{\{1,2\},\ \{1,3\},\ \{2,3,4\}\}$.

approach, the method of maximum a posteriori (MAP) estimate is useful to detect the positive clones. Let $S = (S_1, \ldots, S_m)$ be the random variable for the observation from all pools, and

$$p(X = x|S = s) = p(x|s)$$

be the posterior probability of $X = x = (x_1, \ldots, x_n)$ given $S = s = (s_1, \ldots, s_m)$, where m is the number of pools. The label pattern $x \in \{0,1\}^n$ maximizing the posterior $p(x|s)$ will provide the set of clones which are likely to be positive.

We represent the posterior $p(x|s)$ by $p(x)$ and $p(s|x)$. Using the Bayes formula, we can represent the posterior probability $p(x|s)$ as

$$p(x|s) \propto p(s|x)p(x).$$

For the distinct pools $r, r' \in \mathcal{G}$, the observations s_r and $s_{r'}$ are conditionally independent for given x. Hence the probability $p(s|x)$ is decomposed into the conditional probabilities of $r \in \mathcal{G}$, and then we have

$$p(s|x) = \prod_{r\in\mathcal{G}} p(s_r|x).$$

For each observation $s \in \{0,1,2,3\}$, the conditional probability $p(S_r = s|x)$ is written as

$$p(S_r = s|x) \propto \exp\{c(s,x)\}$$

as the function of s, where $c(s,x)$ is a real-valued function. When we compute the posterior probability of $p(x|s)$, the observations $s_r, r \in \mathcal{G}$ are regarded as constants, and thus $c(s_r, x)$ is written as $c_r(x)$ as the function of the label pattern $x \in \{0,1\}^n$. Note that $c_r(x)$ depends only on z_r which is a realized value of Z_r defined in (1). Then, the posterior probability $p(x|s)$ is given as

$$p(x|s) \propto \exp\Big\{h^\top x + \sum_{r\in\mathcal{G}} c_r(x)\Big\}. \tag{2}$$

Suppose that the parameter h and the functions $c_r(x),\ r \in \mathcal{G}$ are known or these are estimated with satisfactory accuracy.

Table 1: An example of marginal posterior probabilities. The pooling design in Figure 1 and the observation probability $p(S_r = s|x)$ shown in Table 3 are used, and the marginal probability $p(X_i = 1)$ is set to 0.1 for all clones.

$(s_1, s_2, s_3) \in \{0, 1, 2, 3\}^3$	$p_1(X_1 = 1\|s)$	$p_2(X_2 = 1\|s)$	$p_3(X_3 = 1\|s)$	$p_4(X_4 = 1\|s)$
$(3, 0, 0)$	0.043	0.047	0.001	0.011
$(2, 2, 0)$	0.853	0.019	0.019	0.009
$(0, 1, 3)$	0.020	0.016	0.760	0.180
$(0, 0, 3)$	0.001	0.027	0.027	0.429

In general the maximization of $p(x|s)$ in (2) over $x \in \{0, 1\}^n$ is computationally hard unless the set of pools $\mathcal{G}$ has some special property (Pearl, 1988; Cowell et al., 2007). Thus, we take another approach. The marginal probability of x_i for $p(x|s)$ is denoted as $p_i(x_i|s)$, that is

$$p_i(x_i|s) = \sum_{x' \in \{0,1\}^n : x'_i = x_i} p(x'|s) \tag{3}$$

We think that the clones having large marginal posterior $p_i(X_i = 1|s)$ will be positive. Using a threshold for the marginal posterior, we will be able to detect the set of positive clones. As an example Table 1 shows exact marginal posterior probabilities $p_i(X_i = 1|s)$. The pooling design in Figure 1 and the observation probability $p(S_r = s|x)$ shown in Table 3 are used, and the marginal probability $p(X_i = 1)$ is set to 0.1 for all clones. We see that the marginal posterior will be useful to detect the positive clones.

The computation of the marginal posterior is still hard, since there are exponentially many summands in (3). Despite this, we can compute an approximate posterior probability by applying so-called *loopy belief propagation* (loopy BP) algorithm. The details of loopy BP is briefly introduced in Section 3.

3 Loopy Belief Propagation for Computation of Marginal Probability

Loopy belief propagation is a method of computing an approximate marginal probability, which is very useful in stochastic reasoning (Pearl, 1988; Cowell et al., 2007). Let $q(x)$ be a joint probability of high dimensional binary variable $x = (x_1, \dots, x_n) \in \{0, 1\}^n$. In the group test $q(x)$ corresponds to the posterior probability $p(x|s)$. The computation of the marginal $q_i(x_i)$ involves exponentially many sums. To reduce the computational cost, we approximate the joint probability $q(x)$ by a tractable one. Suppose that $q(x)$ is represented by the form of (2), that is,

$$q(x) \propto \exp\left\{h^\top x + \sum_{r \in \mathcal{G}} c_r(x)\right\} \tag{4}$$

and we use the model

$$\bar{q}(x; \theta) \propto \exp\left\{h^\top x + \theta^\top x\right\} \tag{5}$$

to approximate $q(x)$, where $\theta = (\theta_1, \dots, \theta_n)^\top \in \mathbb{R}^n$ is an n dimensional column vector. The parameter θ is determined such that the function $\theta^\top x$ is close to $\sum_{r \in \mathcal{G}} c_r(x)$ up to additive constant. Then, the marginal probability of $q(x)$ will be approximately given by

$$\bar{q}_i(x_i; \theta) \propto \exp\left\{h_i x_i + \theta_i x_i\right\}, \quad i = 1, \dots, n.$$

As a result, we can obtain an approximate value of the marginal probability for $q(x)$. The loopy BP algorithm provides an efficient method of computing the parameter θ. Suppose that θ is decomposed into the sum of parameters $\xi_r \in \mathbb{R}^n,\ r \in \mathcal{G}$, that is,

$$\theta = \sum_{r \in \mathcal{G}} \xi_r.$$

We suppose that the function $c_r(x)$ is approximated by $\xi_r^\top x$ for each pool $r \in \mathcal{G}$. When the parameters $\xi_r,\ r \in \mathcal{G}$ are obtained in mid-flow of the algorithm, we show how to update these parameters. Let ζ_r be defined as

$$\zeta_r = \sum_{s \in \mathcal{G}} \xi_s - \xi_r = \theta - \xi_r.$$

When the function $c_r(x)$ is approximated by $\xi_r^\top x$, the probability $q(x)$ is also approximated by $\exp\{h^\top x + \zeta_r^\top x + c_r(x)\}$. We seek the parameter $\bar{\xi}_r$ such that $\exp\{h^\top x + \zeta_r^\top x + \bar{\xi}_r^\top x\}$ approximates $\exp\{h^\top x + \zeta_r^\top x + c_r(x)\}$ up to the normalization constant. The Kullback-Leibler divergence

$$\mathrm{KL}(p, q) = \sum_{x \in \{0,1\}^n} p(x) \log \frac{p(x)}{q(x)}$$

is used as the discrepancy measure between two probabilities p and q over $\{0, 1\}^n$. We consider the following optimization problem:

$$\begin{aligned}
&\min_{\xi_r \in \mathbb{R}^n} \mathrm{KL}(p, q) \\
&\quad \text{subject to } \ p(x) \propto \exp\{h^\top x + \zeta_r^\top x + c_r(x)\}, \quad q(x) \propto \exp\{h^\top x + \zeta_r^\top x + \xi_r^\top x\}.
\end{aligned}$$

By some calculation, we see that the above problem is represented as the following form:

$$\max_{\xi_r \in \mathbb{R}^n} \sum_{x \in \{0,1\}^n} \exp\{h^\top x + \zeta_r^\top x + c_r(x)\} \cdot \left\{\xi_r^\top x - \sum_{i=1}^{n} \log(1 + \exp\{h_i + \zeta_{ri} + \xi_{ri}\})\right\}. \qquad (6)$$

There are $2^{|r|}$ summands in the function to be optimized, where $|r|$ denotes the number of elements in the set r. When the size of the pool r is not large, the objective function in the optimization problem above is tractable. The parameter ξ_r is updated to $\bar{\xi}_r$ which is the optimal solution of (6). In the same way, the parameters $\xi_r, r \in \mathcal{G}$ and the sum $\theta = \sum_{r \in \mathcal{G}} \xi_r$ are updated sequentially. The convergent point of θ is the output of the algorithm, and we obtain the approximated marginal probability $\bar{q}_i(x_i; \theta), i = 1, \ldots, n$. The loopy BP algorithm is very useful in practice, though the convergence property of the algorithm is not theoretically guaranteed under general condition.

In the literature of DNA library screening, the function $c_r(x)$ depends on the value of $z_r = \max_{i \in r} x_i$, and thus we define

$$c_r(x) = \begin{cases} c_{r1} & z_r = 1, \\ c_{r0} & z_r = 0. \end{cases}$$

Then, the objective function of (6) has a simple form, and the updated parameter $\bar{\xi}_r$ in the loopy BP algorithm is explicitly obtained. See Uehara and Jimbo (2009) for details.

4 Bias of Loopy Belief Propagation

According to Ikeda et al. (2004a) we show the bias introduced by the loopy BP algorithm in the general setup. For each pool $r \in \mathcal{G}$, let $c_r(x)$ be any real-valued function depending only on $x_i, i \in r$, and $q(x)$ be a probability on $\{0,1\}^n$ defined as the form of (4). Let $q_i(x_i)$ be the marginal probability of $q(x)$. We use the statistical model (5) to approximate the joint probability $q(x)$. Let $\bar{q}(x;\theta_0) \propto \exp\{h^\top x + \theta_0^\top x\}$ be the convergent joint probability computed by the loopy BP algorithm applied to $q(x)$. Usually the estimated marginal $\bar{q}_i(x_i;\theta_0)$ is not equal to the true marginal probability $q_i(x_i)$, and the difference $q_i(x_i) - \bar{q}_i(x_i;\theta_0)$ is said to be *bias*. Ikeda et al. (2004a) have analyzed the bias of the loopy BP algorithm, and obtained the asymptotic formula such that

$$q_i(1) - \bar{q}_i(1;\theta_0) \;=\; \frac{1}{2}\sum_{\substack{r,s\in\mathcal{G}\\ r\neq s}} B_{rsi} \;+ \text{(higher order terms)}, \tag{7}$$

where B_{rsi} is related to a geometrical curvature of statistical model $\bar{q}(x;\theta)$.

To show the definition of B_{rsi}, we need to define the matrices $g_{ij}, g_{ir}, \widetilde{g}_{ir}$ and the third order tensor T. Let $\bar{x}_i$ and $\bar{c}_r$ be the expectation of x_i and $c_r(x)$ under $\bar{q}(x;\theta_0)$, that is

$$\bar{x}_i = \sum_{x_i} x_i \bar{q}_i(x_i;\theta_0), \qquad \bar{c}_r = \sum_x c_r(x)\bar{q}(x;\theta_0).$$

Note that the expectation $\bar{x}_i$ is equal to $q_i(1;\theta_0)$, since x_i is the binary variable. The matrix $g_{ij}, i,j = 1,\dots,n$ is the Fisher information matrix of the model $\bar{q}(x;\theta)$ at $\theta = \theta_0$,

$$g_{ij} = \sum_{x_i,x_j}(x_i - \bar{x}_i)(x_j - \bar{x}_j)\bar{q}(x;\theta_0) = \delta_{ij}\sum_{x_i}(x_i - \bar{x}_i)^2\bar{q}_i(x_i;\theta_0) = \delta_{ij}\bar{x}_i(1-\bar{x}_i),$$

where δ_{ij} is the Kronecker's delta function such that $\delta_{ij} = 1$ for $i = j$ and otherwise $\delta_{ij} = 0$. Likewise the matrix g_{ir} for $i = 1,\dots,n,\ r \in \mathcal{G}$ is defined by

$$g_{ir} = \sum_x (x_i - \bar{x}_i)(c_r(x) - \bar{c}_r)\bar{q}(x;\theta_0),$$

and let $\widetilde{g}_{ir}$ be $\widetilde{g}_{ir} = g_{ir}/g_{ii}$. Moreover let the third tensor T be

$$\begin{aligned}
T_{ijk} &= \sum_x (x_i - \bar{x}_i)(x_j - \bar{x}_j)(x_k - \bar{x}_k)\bar{q}(x;\theta_0), \quad i,j,k = 1,\dots,n,\\
T_{ijr} &= \sum_x (x_i - \bar{x}_i)(x_j - \bar{x}_j)(c_r(x) - \bar{c}_r)\bar{q}(x;\theta_0), \quad i,j = 1,\dots,n,\ r \in \mathcal{G},\\
T_{irs} &= \sum_x (x_i - \bar{x}_i)(c_r(x) - \bar{c}_r)(c_s(x) - \bar{c}_s)\bar{q}(x;\theta_0), \quad i = 1,\dots,n,\ r,s \in \mathcal{G}.
\end{aligned}$$

Then B_{irs} is defined as

$$B_{rsi} = -T_{irs} - \sum_{j,k=1}^{n} T_{ijk}\widetilde{g}_{jr}\widetilde{g}_{ks} + \sum_{j=1}^{n}(T_{ijr}\widetilde{g}_{js} + T_{ijs}\widetilde{g}_{jr}), \quad i = 1,\dots,n,\ \ r,s \in \mathcal{G}. \tag{8}$$

Once we obtain the approximate joint probability $\bar{q}(x;\theta_0)$, we can compute B_{rsi} without knowing the target probability $q(x)$. Thus, according to (7) the bias is corrected by adding $\sum_{r\neq s} B_{rsi}/2$ to $\bar{q}_i(1;\theta_0)$.

5 Relation between Pooling Design and Bias of Loopy BP Algorithm

We show some properties of B_{rsi} defined in (8). Let $c_r(x)$ be the function on $\{0,1\}^n$ depending only on $x_i, i \in r$, then $c_r(x)$ is represented as the form of

$$c_r(x) \;=\; h_r + \sum_{\ell} b_{r\ell} \prod_{i\in r}(x_i - a_{r\ell i}), \quad h_r,\; b_{r\ell} \in \mathbb{R},\; a_{r\ell i} \in [0,1]. \tag{9}$$

This fact is shown below. Let $r = \{i_1, \ldots, i_{|r|}\} \subset \{1, \ldots, n\}$, and $\bar{x}$ be $\bar{x} = (\bar{x}_1, \ldots, \bar{x}_{|r|}) \in \{0,1\}^{|r|}$, then we define $c_r(x)$ by

$$c_r(x) = \sum_{\bar{x}\in\{0,1\}^{|r|}} b_{\bar{x}} \prod_{j=1}^{|r|} \big(1 - (x_{i_j} - \bar{x}_j)^2\big) = \sum_{\bar{x}\in\{0,1\}^{|r|}} \left[b_{\bar{x}} \prod_{k=1}^{|r|}(2\bar{x}_k - 1)\right] \prod_{j=1}^{|r|} \big(x_{i_j} - (1-\bar{x}_j)\big). \tag{10}$$

The function $c_r(x)$ has the form of (9), and the variable $x \in \{0,1\}^n$ satisfying $x_{i_j} = \bar{x}_j$ for $j = 1, \ldots, |r|$ is mapped to $b_{\bar{x}} \in \mathbb{R}$. By varying $b_{\bar{x}}$ any function over $\{0,1\}^n$ can be represented by the form above. Though the parameter $a_{r\ell i}$ in (9) can be restricted to the binary set $\{0,1\}$, we allow the mild condition $a_{r\ell i} \in [0,1]$ for convenience.

Example 1. *For the group test*

$$c_r(x) = \rho_r \cdot \max_{i\in r} x_i = \rho_r\{1 - (-1)^{|r|} \prod_{i\in r}(x_i - 1)\}$$

is used. For the low density parity check (LDPC) codes the function

$$c_r(x) = \rho_r \cdot \prod_{i\in r}(1 - 2x_i) = \rho_r(-2)^{|r|} \prod_{i\in r}(x_i - 1/2)$$

is exploited. In the above, the coefficient ρ_r determines the intensity contributed from the pool $r \in \mathcal{G}$.

First, we show the condition that the bias vanishes.

Theorem 1. *Let c_r be real-valued function over $\{0,1\}^n$ depending only on the variables x_i, $i \in r$. Let r, s be distinct subsets of $\{1, \ldots, n\}$. Then, for any functions $c_r(x)$, $c_s(x)$ and any $i = 1, \ldots, n$, B_{rsi} vanishes if $|r \cap s| \le 1$.*

Theorem 1 is a direct conclusion of Theorem 7 in Ikeda et al. (2004a). The proof is deferred to appendix A to show the explicit form of B_{rsi}. Let the packing design be the family of sets $\mathcal{G}$ satisfying $|r \cap s| \le 1$ for any $r, s \in \mathcal{G}$, then Theorem 1 denotes that for the packing design the dominant bias term of loopy BP algorithm vanishes. The packing design is used in the design of group test (Uehara and Jimbo, 2009) and also in the LDPC code (MacKay, 1999). It is numerically shown that the accuracy of approximate probability is superior to other designs with $|r \cap s| \ge 2$. In coding theory, lots of designs of low density parity check (LDPC) code have been intensively studied, and the packing design is known as good error-correcting code (Ikeda et al., 2004a; MacKay, 1999). In Theorem 1 these results are extended to any function $c_r, r \in \mathcal{G}$.

We consider the bias term B_{rsi} for $|r \cap s| \ge 2$.

Theorem 2. *Let c_r and c_s be functions with the form of* (9)*, and suppose that there exists a constant C such that the coefficients $b_{r\ell}, b_{s\ell}$ satisfy*

$$\sum_{\ell,\ell'} \left|b_{r\ell} b_{s\ell'}\right| \leq C.$$

Let $\bar{x}_i$ be the expectation of x_i under the probability $\bar{q}(x;\theta_0) \propto \exp\{h^\top x + \theta_0^\top x\}$ and δ be a real number satisfying

$$0 < |\bar{x}_i - a_{r\ell i}| \leq \delta < 1, \quad 0 < |\bar{x}_i - a_{s\ell i}| \leq \delta < 1,$$

for any i, ℓ, r, s. Then, the intensity of B_{rsi} is bounded above as follows:

$$|B_{rsi}| \leq C \cdot \frac{\delta^{|r|+|s|-2}}{2} \left(1 + \frac{1}{4\delta^2}\right)^{|r\cap s|} \tag{11}$$

The proof is shown in appendix B. It is easy to see the right-hand of (11) is increasing function of $\delta > 0$.

Example 2. *The bias term in the group test is shown. The function c_r is defined as $\rho_r(1-\prod_{i\in r}(1-x_i))$ as shown in Example 1. Suppose $\bar{x}_i = \bar{x}$ holds for all $i = 1, \ldots, n$. Then the bias term B_{rsi} for $i \in r \backslash s$ is given as*

$$\begin{aligned} |B_{rsi}| \;&=\; |\rho_r \rho_s| \bar{x}(1-\bar{x})(1-\bar{x})^{|r|+|s|-1} \left\{ \left(1 + \frac{\bar{x}}{1-\bar{x}}\right)^{|r\cap s|} - 1 - |r\cap s| \frac{\bar{x}}{1-\bar{x}} \right\} \\ &\leq\; |\rho_r \rho_s| \frac{(1-\bar{x})^{|r|+|s|-2}}{2} \left(1 + \frac{1}{4(1-\bar{x})^2}\right)^{|r\cap s|}. \end{aligned}$$

The bias B_{rsi} for $i \in r \cap s$ is also computed in the same way. It is verified that $|B_{rsi}|$ vanishes for $|r \cap s| \leq 1$. When $|r|$ and $|s|$ are fixed, minimization of $|r \cap s|$ will contribute to the reduction of the bias.

Example 3. *Let $\bar{x}_i = \bar{x}$ for $i = 1, \ldots, n$. For the LDPC, the function $c_r(x) = \rho_r(-2)^{|r|} \prod_{i\in r}(x_i - 1/2)$ is used. Then, $|B_{rsi}|$ for $i \in r \setminus s$ is given as*

$$\begin{aligned} |B_{rsi}| &= 2|\rho_r \rho_s| \bar{x}(1-\bar{x})|2\bar{x}-1|^{|r|+|s|-1} \left\{ \left(1 + \frac{\bar{x}(1-\bar{x})}{(\bar{x}-1/2)^2}\right)^{|r\cap s|} - 1 - |r\cap s| \frac{\bar{x}(1-\bar{x})}{(\bar{x}-1/2)^2} \right\} \\ &\leq |\rho_r \rho_s| \frac{|2\bar{x}-1|^{|r|+|s|-2}}{2} \left(1 + \frac{1}{(2\bar{x}-1)^2}\right)^{|r\cap s|}, \end{aligned}$$

It is verified that $|B_{rsi}|$ vanishes for $|r \cap s| \leq 1$. When $\bar{x} \neq 1/2$, the bias $|B_{rsi}|$ is increasing in $|r \cap s|$ when the size of pools $|r|$ is fixed. Thus minimization of $|r \cap s|$ is important to reduce the bias.

The dominant bias is represented as the sum of B_{rsi}. We assume that the constants δ and C in Theorem 2 are also upper bounds for any pair of $r, s \in \mathcal{G}$. Suppose that the size of subset is fixed, i.e. $|r| = d$, and let $m = |\mathcal{G}|$. Then, an upper bound of the bias is given as

$$\left|\frac{1}{2}\sum_{r\neq s} B_{rsi}\right| \leq \frac{Cm(m-1)}{2} \cdot \delta^{2d-2} \left(1 + \frac{1}{4\delta^2}\right)^{\max_{r,s:r\neq s} |r\cap s|}.$$

Suppose that C does not significantly depend on the pooling design. Then, the pooling design minimizing $\max_{r,s:r\neq s}|r\cap s|$ will lead a small estimation bias when we use loopy BP algorithm to compute the approximate posterior probability. In the group test C is almost independent of the pooling design, when the size of the pool, $|r|$, is fixed. Indeed we can choose $C=\max_{r,s}|\rho_r\rho_s|$, where ρ_r is not significantly depend on the pooling design. In terms of the minimization of $\max_{r\neq s}|r\cap s|$, we have the following theorem.

Theorem 3. *For fixed integers m, n and d we consider the optimization problem*

$$\min_{\mathcal{G}} \max_{r,s\in\mathcal{G}:r\neq s} |r\cap s|, \quad \text{subject to } |r|=d,\ {}^\forall r\in\mathcal{G}, \tag{12}$$

where $\mathcal{G}$ consists of m subsets of $\{1,\dots,n\}$. Suppose that there exists a pooling design $\bar{\mathcal{G}}$ satisfying the constraint of (12) *and the condition that*

$$\begin{aligned} &i)\ |r\cap s|=\bar\lambda \textit{ or } \bar\lambda-1 \textit{ for all } r,s\in\bar{\mathcal{G}},\ r\neq s,\\ &ii)\ |\{r\in\bar{\mathcal{G}}\mid i\in r\}|=\bar k \textit{ or } \bar k-1 \textit{ for all } i=1,\dots,n, \end{aligned} \tag{13}$$

where $\bar\lambda=\max_{r,s\in\bar{\mathcal{G}}:r\neq s}|r\cap s|$ and $\bar k=\lceil md/n\rceil$. Then the pooling design $\bar{\mathcal{G}}$ is an optimal solution of (12).

Proof. For a fixed pooling design $\mathcal{G}$ let k_i be $k_i=|\{r\in\mathcal{G}\mid i\in r\}|$, that is k_i stands for the number of pools including the clone i. Then we have the equality

$$\sum_{i=1}^n k_i=md, \qquad \sum_{i=1}^n k_i(k_i-1)=\sum_{\substack{r,s\in\mathcal{G}\\ r\neq s}}|r\cap s|.$$

Since the mean value is less than or equal to the maximum value, we have

$$\max_{\substack{r,s\in\mathcal{G}\\ r\neq s}}|r\cap s|\geq\frac{1}{m(m-1)}\sum_{\substack{r,s\in\mathcal{G}\\ r\neq s}}|r\cap s|=\frac{\sum_{i=1}^n k_i^2-md}{m(m-1)}$$

Some calculation leads that the quadratic function $\sum_{i=1}^n k_k^2$ is minimized at $(\bar k_1,\dots,\bar k_n)=(\bar k,\dots,\bar k,\bar k-1,\dots,\bar k-1)$ under the constraint that $\sum_{i=1}^n k_i=md$ for integers $k_1,\dots,k_n$. Thus for any pooling design $\mathcal{G}$, the objective function in (12) is bounded below by $(\sum_{i=1}^n\bar k_i^2-md)/m(m-1)$ which depends only on n, m and d. For the pooling design $\bar{\mathcal{G}}$ satisfying (13), we have

$$\bar\lambda=\max_{\substack{r,s\in\bar{\mathcal{G}}\\ r\neq s}}|r\cap s|\geq\frac{\sum_{i=1}^n\bar k_i^2-md}{m(m-1)}>\bar\lambda-1.$$

The last inequality comes from the facts that $|r\cap s|$ is equal to $\bar\lambda$ or $\bar\lambda-1$ and that there exists a pair r,s such that $|r\cap s|=\bar\lambda$. Thus $\bar{\mathcal{G}}$ is an optimal design, since $\bar{\mathcal{G}}$ attains the least integer which is greater than or equal to the lower bound of the objective function. $\square$

The pooling design called balanced incomplete block design (BIBD) has the property such that in conditions i) and ii) of Theorem 3 equalities $|r\cap s|=\bar\lambda$ and $|\{r\in\mathcal{G}\mid i\in r\}|=\bar k$ always hold. According to Theorem 3, a BIBD is an optimal solution in the sense that it has the maximum possible number of clones n for given number of pools m among the designs satisfying (12) if it exists for specified n, m and d.

A BIBD is often called a 2-design. The existence condition and the construction method of BIBD's have been intensively investigated in the field of combinatorics (Beth et al., 1999; Colbourn and Dinitz, 2007). Among them, constructions based on finite fields and finite geometries are well investigated. Also many recursive constructions or composition methods are developed. Tables of the existing BIBD's for small orders are listed in Chapter 2 of Colbourn and Dinitz (2007). The designs utilized in this paper are constructed based on Theorem 2 in Wilson (1972). See also Lemma 6.3 in Beth et al. (1999) for details.

6 Numerical Experiments

The bias correction is examined in some numerical experiments. In the experiment, we specify the number of clones (n), the number of pools (m) and the size of pool $|r|$, and then construct a pooling design $\mathcal{G}$ satisfying the condition $|r \cap s| = \lambda$ for any pair of pools $r, s \in \mathcal{G}$, where λ is a prespecified constant. Then, the group test is conducted by using the pooling design.

In numerical experiments, the number of clones is set to $n = 24, 1314$ or 1552, and the pooling design is prepared based on the balanced incomplete block design. Table 2 illustrates the pooling design for each simulation. Basically, the same BIB designs are combined to make larger pooling design. In order to build the pooling design such that any pair of clones is not assigned exactly the same pools, we applied randomization technique. The priori probability for each clone is defined as $p_i(X_i = 1) = 0.1$ for $n = 24$ and $p_i(x_i) = 0.002$ for $n = 1314$ and $n = 1552$. As shown in Table 3 the conditional probability of the observation, $p(S_r = s_r|Z_r = z_r)$, has been estimated by the experiments of an actual DNA library screening (Knill et al., 1996), and thus we use the probability in our algorithm.

In the simulation, some positive clones are randomly chosen out of n clones, and the observations $s_r \in \{0, 1, 2, 3\}, r \in \mathcal{G}$ are generated according to the defined probability. The number of positive clones varies from one to four. Then, we estimate the marginal posterior probability $p_i(x_i|s)$. The estimated probability is compared to the true posterior probability computed by the Markov Chain Monte Carlo (MCMC) method Knill et al. (1996). Table 4 shows the estimated result in the descending order of the marginal posterior probability. In both methods almost the same clones are highly placed. Note that the MCMC method is computationally demanding. We use the MCMC method in order to obtain precise posterior probability which is used to assess the estimated (bias-corrected) posterior probability. In the numerical experiments, we use Concave-Convex Procedure (CCCP) algorithm (Yuille, 2002) to compute the posterior probability instead of the conventional loopy BP algorithm. The CCCP has the same extremal solution as the loopy BP algorithm, though the CCCP may have better convergence property. The computation time is shown in Table 5. The CCCP is compared with the MCMC method. Overall CCCP is efficient for large set of clones. We have confirmed that the computation time for bias correction is negligible.

The bias-correction term $\frac{1}{2}\sum_{r \neq s: r,s \in \mathcal{G}} B_{rsi}$ is added to the estimated posterior probability given by CCCP. The accuracy of the estimator is measured by the Kullback-Leibler (KL) divergence. Let $q_i(x_i)$ be the true posterior given by the MCMC method for $n = 1314, 1552$. For $n = 24$, the exact posterior probability is available. the discrepancy between q_i and the estimated posterior $\bar{q}_i$ for $i = 1, \ldots, n$ is measured by

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{x_i \in \{0,1\}} q_i(x_i|s) \log \frac{q_i(x_i|s)}{\bar{q}_i(x_i|s)}.$$

In the numerical simulation we conducted the estimation 1000 times with different random seed, and the KL-divergence is averaged over the repetition.

Table 6, Table 7, and Table 8 show the results for each pooling design. The first column shows the number of positive clones out of n clones, and the second and third columns present the averaged KL-divergence for the estimator given by CCCP and its bias-corrected variant, respectively. When $|r \cap s|$ is less than three, the bias correction works well to improve the accuracy of the estimated posterior as shown in Table 6 and Table 8. Table 7 shows the result using the pooling design satisfying $|r \cap s| = 3$. In this case, the bias-correction does not necessarily improve the estimator. This result indicates that not only the dominant bias term $\frac{1}{2}\sum_{r \neq s} B_{rsi}$ but also the higher order term will be necessary to improve the estimator.

In the simple experiments, the bias correction may be useful to improve the estimated posterior when the pooling design $\mathcal{G}$ satisfies $|r \cap s| = 2$ for $r, s \in \mathcal{G}$.

7 Concluding Remarks

For the pooling design we have proposed the bias corrected estimator of the marginal posterior probability based on the result of Ikeda et al. (2004a,b). We analyzed an upper bound of the bias term and showed that BIB design will make the bias small comparing to other pooling designs. In numerical experiments, the bias correction works well to improve the marginal posterior, even when $|r \cap s| = 2$ holds for the pooling design $\mathcal{G}$. We confirmed that the correction of the dominant bias term does not necessarily improve the estimator, when the pooling design satisfies $|r \cap s| = 3$. Investigating higher order bias correction will be an important future work.

A Calculation of B_{rsi}

Theorem 1 is obtained as a direct conclusion of Theorem 7 in Ikeda et al. (2004a). Here, we compute B_{rsi} to show its explicit form, and verify that $B_{rsi} = 0$ for $|r \cap s| \leq 1$.

As shown in (9) and (10), any function $c_r(x)$ over $\{0,1\}^n$ depending on only $x_i, i \in r$ is represented by the linear sum of the functions having the form of $\prod_{i \in r}(x_i - a_i)$, where $a_i \in [0,1]$. Moreover, the bias term B_{rsi} is bilinear in $c_r(x) - \bar{c}_r$ and $c_s(x) - \bar{c}_s$. Therefore, it is enough to consider the case that c_r and c_s are given as $c_r(x) = \prod_{i \in r}(x_i - a_{ri})$ and $c_s(x) = \prod_{i \in s}(x_i - a_{si})$ for $a_{ri}, a_{si} \in [0,1]$.

Let us define $e_r(r')$ for the subset $r' \subset r$ by

$$e_r(r') = \prod_{i \in r'}(\bar{x}_i - a_{ri}).$$

Let $E[\,\cdot\,]$ be the expectation by the probability $\bar{q}(x;\theta_0)$, then we have $E[c_r] = e_r(r)$. Building blocks for the calculation of the bias term are given as follows. The matrix g_{ij}, g_{ir} and $\widetilde{g}_{ir}$ are given as

$$g_{ii} = \bar{x}_i(1-\bar{x}_i), \quad g_{ir} = \begin{cases} 0 & i \notin r, \\ \bar{x}_i(1-\bar{x}_i)e_r(r \setminus \{i\}) & i \in r, \end{cases}$$

$$\therefore \quad \widetilde{g}_{ir} = \frac{g_{ir}}{g_{ii}} = \begin{cases} 0 & i \notin r, \\ e_r(r \setminus \{i\}) & i \in r. \end{cases}$$

The third tensor T_{ijk} is computed as follows:

$$T_{ijk} = \bar{x}_i(1-\bar{x}_i)(1-2\bar{x}_i)\delta_{ij}\delta_{ik},$$

$$\therefore \quad \sum_{jk} T_{ijk}\widetilde{g}_{jr}\widetilde{g}_{ks} = \begin{cases} \bar{x}_i(1-\bar{x}_i)(1-2\bar{x}_i)e_r(r \setminus \{i\})e_s(s \setminus \{i\}) & i \in r \cap s, \\ 0 & \text{otherwise} \end{cases}$$

Table 2: Balanced in complete block (BIB) designs used in the simulation and the prior probability are shown. In our context the conventional notation (v, r, b, k, λ) for $\mathrm{BIBD}(v, r, b, k, \lambda)$ corresponds to $(m, |r|, n, nm/|r|, |r \cap s|)$ for $r, s \in \mathcal{G}$, $r \neq s$, where $|r|$ and $|r \cap s|$ take a constant number. The identical BIB designs are combined to make larger pooling design. When the base design is $\mathrm{BIBD}(v, r, b, k, \lambda)$ and the repetition is t, the pooling design defined from $\mathrm{BIBD}(v, r \cdot z, b \cdot t, k, \lambda \cdot t)$ is constructed by combining the base design. In order to build the pooling design such that any pair of clones is not assigned exactly the same pools, we applied randomization technique.

#clones	base design	repetition	prior: $p_i(X_i = 1)$
$n = 24 = 12 \times 2$	$\mathrm{BIBD}(9, 4, 12, 3, 1)$	2	0.1
$n = 1314 = 438 \times 3$	$\mathrm{BIBD}(73, 24, 438, 4, 1)$	3	0.002
$n = 1552 = 776 \times 2$	$\mathrm{BIBD}(97, 32, 776, 4, 1)$	2	0.002

Table 3: The conditional probability estimated by the experiments of an actual DNA library screening (Knill et al., 1996).

$P(S_r = 0 \mid Z_r = 0) = 0.871$	$P(S_r = 0 \mid Z_r = 1) = 0.05$
$P(S_r = 1 \mid Z_r = 0) = 0.016$	$P(S_r = 1 \mid Z_r = 1) = 0.11$
$P(S_r = 2 \mid Z_r = 0) = 0.035$	$P(S_r = 2 \mid Z_r = 1) = 0.27$
$P(S_r = 3 \mid Z_r = 0) = 0.078$	$P(S_r = 3 \mid Z_r = 1) = 0.57$

Table 4: Estimated posterior probability in the preliminary experiments. The estimated probability using loopy BP algorithm is compared to the true posterior computed by the Markov Chain Monte Carlo (MCMC) method Knill et al. (1996). The result is shown in the descending order of the marginal posterior probability. In both methods almost the same clones are highly placed.

	loopy BP		MCMC	
rank	clone id	posterior	clone id	posterior
1	336	0.8393	336	0.8345
2	768	0.0615	768	0.0628
3	125	0.0574	125	0.0608
4	764	0.0419	81	0.0400
5	81	0.0409	764	0.0382

Table 5: The computation time (second) is shown. The CCCP is compared with the MCMC method. Overall CCCP is efficient for the large set of clones. We have confirmed that the computation time for bias correction is negligible.

n	CCCP	MCMC
981	0.22	1.91
1298	0.27	2.49
3088	0.81	8.49
6371	1.68	17.60
10121	3.33	27.73
30050	11.09	81.80

Table 6: The numerical results for pooling design such that $n = 24,\ m = 9,\ |r| = 8,\ |r \cap s| = 2$ are shown. The prior probability is set to $p_i(X_i = 1) = 0.1$ for all $i = 1, \dots, n$. The first column shows the number of positive clones out of n clones, and the second and third column presents the averaged KL-divergence for the CCCP and its bias-corrected variant from the posterior given by the MCMC method, respectively.

# positive	CCCP	bias-corrected CCCP
1	11.67e-04	6.67e-04
2	10.57e-04	6.01e-04
3	7.020e-04	4.26e-04
4	4.160e-04	2.76e-04

Table 7: The numerical results for pooling design such that $n = 1314,\ m = 73,\ |r| = 72,\ |r \cap s| = 3$ are shown. The prior probability is set to $p_i(X_i = 1) = 0.002$ for all $i = 1, \dots, n$.

# positive	CCCP	bias-corrected CCCP
1	3.80 e-05	2.40 e-05
2	1.80 e-05	1.80 e-05
3	10.1 e-05	14.3 e-05
4	4.80 e-05	5.20 e-05

Table 8: The numerical results for pooling design such that $n = 1552,\ m = 97,\ |r| = 64,\ |r \cap s| = 2$ are shown. The prior probability is set to $p_i(X_i = 1) = 0.002$ for all $i = 1, \dots, n$.

# positive	CCCP	bias-corrected CCCP
1	0.90 e-05	0.90 e-05
2	1.70 e-05	1.50 e-05
3	3.30 e-05	1.90 e-05
4	2.80 e-05	2.60 e-05

For T_{ijr} we see that $T_{ijr} = 0$ when $i \notin r$ or $j \notin r$ holds. Then, we compute T_{ijr} for $i, j \in r$. When $i = j \in r$ holds, we have

$$\begin{aligned} T_{iir} &= E[(x_i - \bar{x}_i)^2(c_r(x) - \bar{c}_r)] \\ &= E[(x_i - \bar{x}_i)^2(x_i - a_{ri})]e_r(r \setminus \{i\}) - \bar{x}_i(1 - \bar{x}_i)e_r(r) \\ &= \bar{x}_i(1 - \bar{x}_i)(1 - 2\bar{x}_i)e_r(r \setminus \{i\}). \end{aligned}$$

In the same way, we obtain

$$T_{ijr} = \begin{cases} \bar{x}_i(1 - \bar{x}_i)(1 - 2\bar{x}_i)e_r(r \setminus \{i\}) & i = j \in r \\ \bar{x}_i(1 - \bar{x}_i)\bar{x}_j(1 - \bar{x}_j)e_r(r \setminus \{i, j\}) & i, j \in r, i \neq j \end{cases}$$

Note that

$$\sum_j (T_{ijr}\widetilde{g}_{js} + T_{ijs}\widetilde{g}_{jr}) = \sum_{j \in r \cap s} (T_{ijr}\widetilde{g}_{js} + T_{ijs}\widetilde{g}_{jr})$$

holds. For $i \notin r \cup s$, the equality $T_{ijs} = T_{ijr} = 0$ holds, and for $r \cap s = \emptyset$ we have $T_{ijs}\widetilde{g}_{jr} = 0$. Thus,

$$i \notin r \cup s \ \text{ or } \ r \cap s = \emptyset \implies \sum_j (T_{ijr}\widetilde{g}_{js} + T_{ijs}\widetilde{g}_{jr}) = 0$$

holds. Then for $r \cap s \neq \emptyset$ we consider other two cases: $i \in r \setminus s$ (or $i \in s \setminus r$) and $i \in r \cap s$. Some calculation leads to

$$\begin{aligned} i \in r \backslash s: \ & \sum_{j \in r \cap s} (T_{ijr}\widetilde{g}_{js} + T_{ijs}\widetilde{g}_{jr}) \ = \ \bar{x}_i(1 - \bar{x}_i) \sum_{j \in r \cap s} \bar{x}_j(1 - \bar{x}_j)e_r(r \backslash \{i, j\})e_s(s \backslash \{j\}), \\ i \in r \cap s: \ & \sum_{j \in r \cap s} (T_{ijr}\widetilde{g}_{js} + T_{ijs}\widetilde{g}_{jr}) \\ & = 2\bar{x}_i(1 - \bar{x}_i)(1 - 2\bar{x}_i)e_r(r \backslash \{i\})e_s(s \backslash \{i\}) \\ & \quad + \bar{x}_i(1 - \bar{x}_i) \sum_{j \in r \cap s \backslash \{i\}} \bar{x}_j(1 - \bar{x}_j)\big[e_r(r \backslash \{i, j\})e_s(s \backslash \{j\}) + e_s(s \backslash \{i, j\})e_r(r \backslash \{j\})\big]. \end{aligned}$$

For T_{irs}, we see that

$$i \notin r \cup s \ \text{ or } \ r \cap s = \emptyset \implies T_{irs} = 0$$

holds. Under the condition that $r \cap s \neq \emptyset$, we consider other two cases: $i \in r \setminus s$ and $i \in r \cap s$,

$$\begin{aligned} & i \in r \backslash s: \\ & \quad T_{irs} = \bar{x}_i(1 - \bar{x}_i)\Bigg[e_r(r \backslash (s \cup \{i\}))e_s(s \backslash r) \prod_{j \in r \cap s} \{\bar{x}_j(1 - \bar{x}_j) + (\bar{x}_j - a_{rj})(\bar{x}_j - a_{sj})\} \\ & \qquad\qquad - e_r(r \backslash \{i\})e_s(s)\Bigg], \\ & i \in r \cap s: \\ & \quad T_{irs} = \bar{x}_i(1 - \bar{x}_i)\Bigg[(1 - a_{ri} - a_{si})e_r(r \backslash s)e_s(s \backslash r) \prod_{j \in r \cap s \backslash \{i\}} \{\bar{x}_j(1 - \bar{x}_j) + (\bar{x}_j - a_{rj})(\bar{x}_j - a_{sj})\} \\ & \qquad\qquad - e_r(r)e_s(s \backslash \{i\}) - e_r(r \backslash \{i\})e_s(s)\Bigg]. \end{aligned}$$

We show B_{rsi} below. Remember that

$$B_{rsi} = -T_{irs} - \sum_{j,k=1}^{n} T_{ijk}\widetilde{g}_{jr}\widetilde{g}_{ks} + \sum_{j=1}^{n}(T_{ijr}\widetilde{g}_{js} + T_{ijs}\widetilde{g}_{jr}).$$

Then, we have

$$r \cap s = \emptyset \text{ or } i \notin r \cap s \implies B_{rsi} = 0,$$

since all terms vanish. To compute other cases, we use the formula,

$$\begin{aligned}&\prod_{j\in r\cap s}\{\bar{x}_j(1-\bar{x}_j) + (\bar{x}_j - a_{rj})(\bar{x}_j - a_{sj})\}\\ &= \sum_{k=0}^{|r\cap s|}\sum_{\substack{v\subset r\cap s\\ |v|=k}} e_r((r\cap s)\setminus v)\cdot e_s((r\cap s)\setminus v)\cdot\prod_{j\in v}\bar{x}_j(1-\bar{x}_j),\end{aligned}$$

and so forth. Then B_{rsi} is represented as follows,

$$\begin{aligned}i \in r\backslash s:\ B_{rsi} &= -\bar{x}_i(1-\bar{x}_i)\sum_{k=2}^{|r\cap s|}\sum_{\substack{v\subset r\cap s\\ |v|=k}} e_r(r\backslash(v\cup\{i\}))e_s(s\setminus v)\prod_{j\in v}\bar{x}_j(1-\bar{x}_j),\\ i\in r\cap s:\ B_{rsi} &= \bar{x}_i(1-\bar{x}_i)\Bigg[(2\bar{x}_i - 1)\sum_{k\in r\cap s\backslash\{i\}}\bar{x}_k(1-\bar{x}_k)e_r(r\backslash\{i,k\})e_s(s\backslash\{i,k\})\\ &\quad - (1-a_{ri}-a_{si})\sum_{k=2}^{|r\cap s|-1}\sum_{\substack{v\subset r\cap s\backslash\{i\}\\ |v|=k}} e_r(r\backslash(\{i\}\cup v))e_s(s\backslash(\{i\}\cup v))\prod_{j\in v}\bar{x}_j(1-\bar{x}_j)\Bigg].\end{aligned}$$

Paying attention to the summation, we see that B_{rsi} vanishes for $|r\cap s| \leq 1$.

B Upper bound of $|B_{rsi}|$

First we derive an upper bound of $|B_{rsi}|$ for $c_r(x) = \prod_{i\in r}(x_i - a_{ri})$ and $c_s(x) = \prod_{i\in s}(x_i - a_{si})$. Suppose that there exists δ such that $0 < |\bar{x}_i - a_i| \leq \delta < 1$ holds for all $i = 1,\ldots,n$ and all a_i appearing in c_r and c_s. Then, we obtain an upper bound of $|B_{rsi}|$. We use $\bar{x}_i(1-\bar{x}_i) \leq 1/4$. For $i \in r\backslash s$ we have

$$|B_{rsi}| \leq \frac{1}{4}\sum_{k=2}^{|r\cap s|}\sum_{\substack{v\subset r\cap s\\ |v|=k}}\delta^{|r|-1-k}\delta^{|s|-k}\left(\frac{1}{4}\right)^k \leq \frac{\delta^{|r|+|s|-1}}{4}\left(1+\frac{1}{4\delta^2}\right)^{|r\cap s|}$$

and in the same way for $i \in r\cap s$ we obtain

$$|B_{rsi}| \leq \frac{\delta^{|r|+|s|-2}}{4}\left[\frac{|r\cap s|-1}{4\delta^2} + \left(1+\frac{1}{4\delta^2}\right)^{|r\cap s|-1}\right] \leq \frac{\delta^{|r|+|s|-2}}{2}\left(1+\frac{1}{4\delta^2}\right)^{|r\cap s|-1}.$$

Therefore, for any case we have

$$|B_{rsi}| \leq \frac{\delta^{|r|+|s|-2}}{2}\left(1+\frac{1}{4\delta^2}\right)^{|r\cap s|}.$$

Next we suppose that

$$c_r(x) = h_r + \sum_{\ell} b_{r\ell} \prod_{i\in r}(x_i - a_{r\ell i}),$$
$$c_s(x) = h_s + \sum_{\ell'} b_{s\ell'} \prod_{i'\in s}(x_{i'} - a_{s\ell' i'}).$$

Since B_{rsi} is bilinear in $c_r(x) - \bar{c}_r$ and $c_s(x) - \bar{c}_s$, we have

$$|B_{rsi}| \leq \sum_{\ell,\ell'} \left|b_{r\ell} b_{s\ell'}\right| \cdot \frac{\delta^{|r|+|s|-2}}{2}\left(1+\frac{1}{4\delta^2}\right)^{|r\cap s|} \leq C \cdot \frac{\delta^{|r|+|s|-2}}{2}\left(1+\frac{1}{4\delta^2}\right)^{|r\cap s|}.$$

References

Beth T., Jungnickel D., Lenz H., 1999. *Design Theory*. Cambridge Univ. Press.

Bruno W., Knill E., Balding D., Bruce D., Doggett N., Sawhill W., Stallings R., Whittaker C., Torney D., 1995. Efficient pooling designs for library screening. *Genomics*, 26(1), 21–30.

Colbourn C.J., Dinitz J.H. (eds.), 2007. *Handbook of combinatorial designs*. CRC Press, Boca Raton, FL, second edition.

Cowell R.G., Dawid A.P., Lauritzen S.L., Spiegelhalter D.J., 2007. *Probabilistic Networks and Expert Systems: Exact Computational Methods for Bayesian Networks*. Springer Publishing Company, Incorporated.

Du D.Z., Hwang F.K., 1999. *Combinatorial Group Testing and Its Applications, 2nd ed.* World Scientific.

Ikeda S., Tanaka T., Amari S., 2004a. Information geometry of turbo and low-density parity-check codes. *IEEE Transactions on Information Theory*, 50(6), 1097–1114.

Ikeda S., Tanaka T., Amari S., 2004b. Stochastic reasoning, free energy, and information geometry. *Neural Comput.*, 16(9), 1779–1810.

Knill E., Schliep A., Torney D., 1996. Interpretation of pooling experiments using the markov chain monte carlo method. *J. Computational Biology*, 3(3), 395–406.

MacKay D.J.C., 1999. Good error correcting codes based on very sparse matrices. *IEEE Transactions on Information Theory*, 45(2), 399–431.

Mézard M., Toninelli C., 2007. Group testing with random pools: Optimal two-stage algorithms. *arXiv:0706.3104*.

Ngo H., Du D., 2000. A survey on combinatorial group testing algorithms with applications to dna library screening. *Discrete Math. Problems with Medical Applications, DIMACS Ser. Discrete Math. and Theoretical Computer Science*, 55, 171–182.

Pearl J., 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. ISBN 1558604790.

Richardson T.J., Shokrollahi M.A., Urbanke R.L., 2001. Design of capacity-approaching irregular low-density parity-check codes. *IEEE Transactions on Information Theory*, 47(2), 619–637.

Uehara H., Jimbo M., 2009. A positive detecting code and its decoding algorithm for dna library screening. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 6(4), 652–666.

Wilson R.M., 1972. Cyclotomy and difference families in elementary abelian groups. *J. Number Theory*, 4, 17–47.

Wu W., Li Y., Huang C., D.Z.Du, 2004. Molecular biology and pooling design. *Proc. Workshop Data Mining in Biomedicine (DMB '04)*.

Yuille A.L., 2002. CCCP algorithms to minimize the bethe and kikuchi free energies: convergent alternatives to belief propagation. *Neural Comput.*, 14(7), 1691–1722. ISSN 0899-7667.